

# LAB GUIDE

---

## Lab: Implementing Containers on Azure VMs.

### Pre-requisites

- Microsoft Azure Account: You'll need a valid and active Azure account for the Azure labs.

### Length

30 minutes

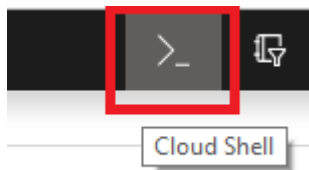
### Before you begin

1. Enter in the following link [Portal Azure](#) and follow the next instructions

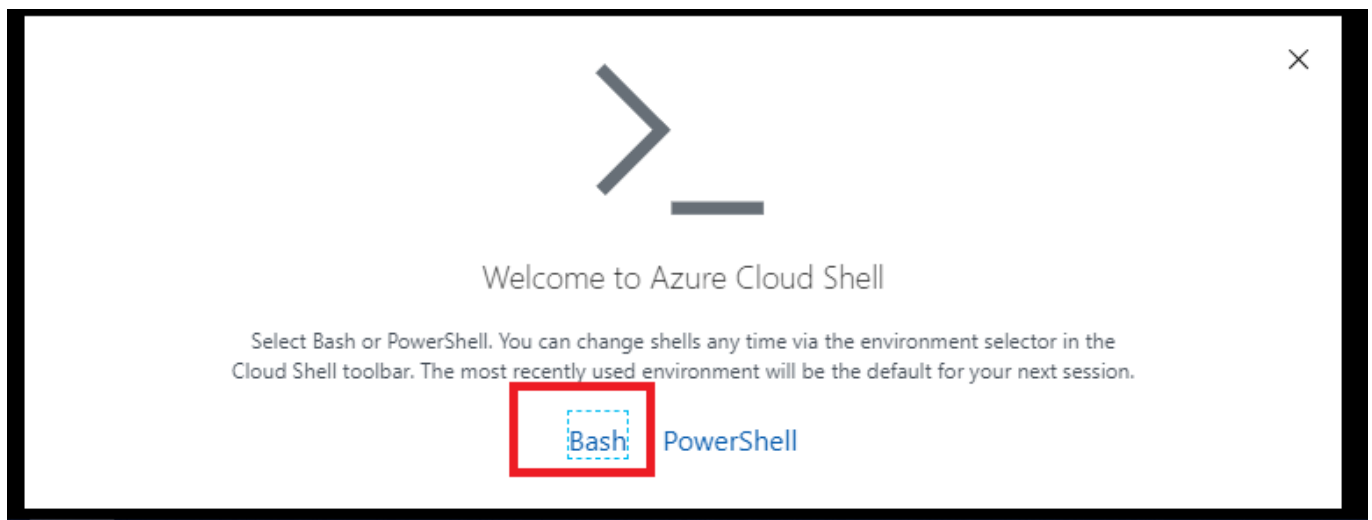
## Exercise 1: Implementing Docker hosts on Azure VMS

### Task 1: Connect to Azure Cloud Shell

1. In the Azure portal click the Cloud Shell icon.



2. If you are presented with the Welcome to Azure Cloud Shell pane, click Bash (Linux). (If you've used the Cloud Shell before and it defaults to PowerShell, click the PowerShell drop-down and select Bash. Continue to Task 2)



3. If you are presented with the You have no storage mounted message, click Show advanced settings

You have no storage mounted

Azure Cloud Shell requires an Azure file share to persist files. [Learn more](#)  
 This will create a new storage account for you and this will incur a small monthly cost. [View pricing](#)

\* Subscription  
 Pase para Azure: patrocinio [Show advanced settings](#)

[Create storage](#) [Close](#)

4. In the resulting pane, specify the following settings and click Create storage:

- Subscription: ensure that the name of the target Azure subscription appears in the drop-down list
- Cloud Shell region: select the name of the Azure region that is available in your subscription and which is closest to the lab location
- Resource group: Select Use Existing and select the group Module-02-XXXX Where XXXX is a number generated for this lab.
- Storage account: ensure that Create new is selected and type a unique name of between 3 and 24 characters consisting of lower-case letters and digits
- File share: ensure that Create new is selected and type a unique name of between 3 and 63 characters consisting of lower-case letters, digits and dashes.

You have no storage mounted

\* Subscription  
 Pase para Azure: patrocinio

\* Cloud Shell region  
 East US [Hide advanced settings](#)

\* Resource group  
☒ Create new ☐ Use existing  
 mymodule2RG

\* Storage account  
☒ Create new ☐ Use existing  
 mystrmodule2

\* File share  
☒ Create new ☐ Use existing  
 myfsmodule2

*Storage accounts are filtered for your selected Cloud Shell region and LRS/GRS/ZRS account types.*

[Create storage](#) [Close](#)

5. Wait for the deployment to be completed.

## Task 2: Create a new azure VM running Docker

1. In the azure portal, from the cloud shell, Deploy a template that will create a new Azure VM hosting Docker by typing the following command and pressing Enter, change the XXXXXX on the resource group to match with the number obtained from the lab environment, which will be on your credentials between odl\_user\_ and @:

```
az group deployment create --resource-group Module-02-XXXXX --template-uri
https://raw.githubusercontent.com/Azure/azure-quickstart-templates/master/docker-
simple-on-ubuntu/azuredeploy.json
```

2. The terminal request the following values

```
+ adminUsername: student
+ adminPassword: Pa55word234234
+ dnsNameForPublicIP: any valid, unique name consisting of lowercase letters
and digits
```

3. Wait for the deployment to be completed.

## Exercise 2: Deploying containers to Azure VMs

### Task 1: Connect to an Azure VM running Docker

1. In the Azure portal, in the Cloud Shell pane, identify the fully qualified domain name, again, please change the XXXXXX with the number generated for your lab.

```
FQDN=$(az vm show --resource-group Module-02-XXXXX --name myDockerVM --show-
details --query [fqdns] --output tsv)
```

2. From the cloud shell identify the fully qualified name via which you can access the Azure VM by typing the following and pressing Enter:

```
echo $FQDN
```

3. From the Cloud Shell pane, establish an SSH session to the Azure VM by typing the following and pressing Enter:

```
ssh student@$FQDN
```

4. The Cloud Shell will display a message informing you that the authenticity of the remote host cannot be established. This is expected. Type yes and press Enter to continue connecting
5. When prompted for the password, type Pa55word234234 and then press Enter. You should be presented with the student@MyDockerVM prompt.

### Task 2: Deploy a container to a Docker host running on an Azure VM

1. From the Cloud shell pane, within the SSH session to the Azure VM running Docker enter this command:

```
docker run -d -p 80:80 --restart=always nginx
```

**Note:** To configure the restart policy for a container, use the `--restart` flag when using the `docker run` command. The value `always`, Always restart the container if it stops.

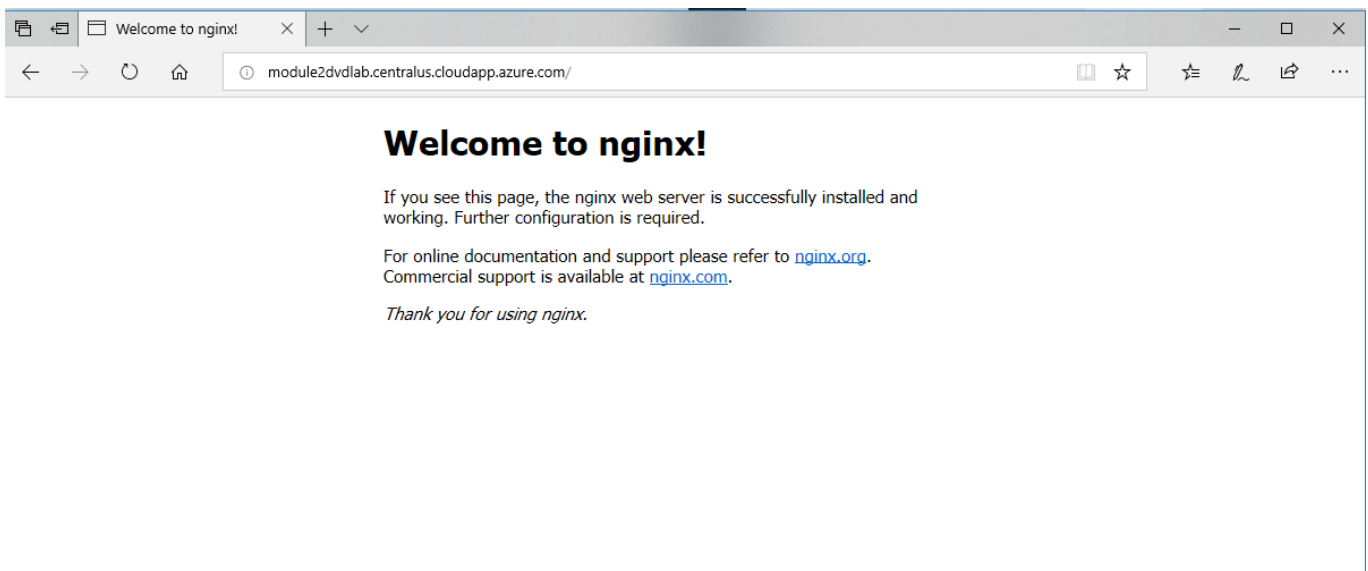
```
student@MyDockerVM:~$ docker run -d -p 80:80 --restart=always nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
6ae821421a7d: Pull complete
da4474e5966c: Pull complete
eb2aec2b9c9f: Pull complete
Digest: sha256:dd2d0ac3fff2f007d99e033b64854be0941e19a2ad51f174d9240dda20d9f534
Status: Downloaded newer image for nginx:latest
185d0f3d86a97d9678525079eb1cd1752da436f73f691ad309f3c739dabb76d8
```

2. Check the process of the running container deployment with this command:

```
docker ps
```

```
student@MyDockerVM:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
PORTS              NAMES
185d0f3d86a9        nginx              "nginx -g 'daemon of..." 25 seconds ago     Up 23 seconds
0.0.0.0:80->80/tcp   eager_nightingale
```

3. Start Microsoft Edge (or any browser) and browse to the URL matching the DNS fully qualified domain name you obtained in the previous task. Verify that the browser displays the Welcome to nginx! page



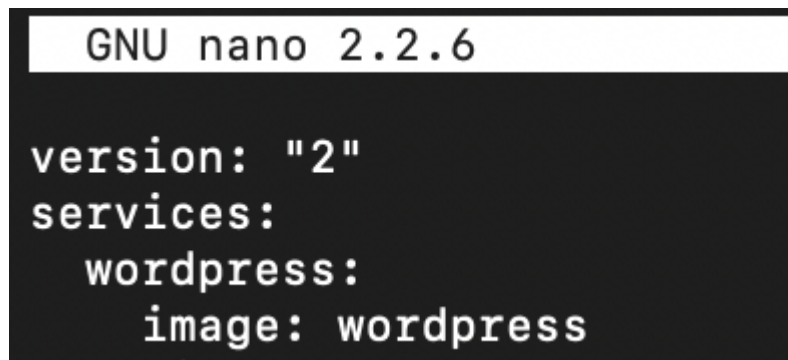
Once you completed this exercise, you have successfully run a sample containerized web server nginx on the Docker host Azure VM.

## Exercise 3: Deploying multi-container applications to Azure VMs with Docker Compose

## Task 1: Create a compose file

1. From the Cloud shell pane, within the SSH session to the Azure VM running Docker create a docker-compose.yml file by typing the following and then pressing Enter:

```
nano docker-compose.yml
```



```
GNU nano 2.2.6

version: "2"
services:
  wordpress:
    image: wordpress
```

2. In the nano editor interface, type the following content

```
version: "2"
services:
  wordpress:
    image: wordpress
    links:
      - db:mysql
    ports:
      - 8080:80
  db:
    image: mariadb
    environment:
      MYSQL_ROOT_PASSWORD: yourpassword
```

Note: Be careful when typing the text above. Make sure you include the spaces to the left of the text (each indent should be a multiple of 2 spaces)

3. Once you typed in the text, press the Ctrl+O key combination and then press Enter.
4. Next, press the Ctrl+X key combination to exit the nano editor.

## Task 2: Deploy the containers with docker-compose to an Azure VM

1. From the Cloud shell pane, within the SSH session to the Azure VM running Docker, to deploy multi-container application defined on the previous task, type the following command:

```
docker-compose up -d
```

```

Creating network "student_default" with the default driver
Pulling db (mariadb:latest)...
latest: Pulling from library/mariadb
6cf436f81810: Pull complete
987088a85b96: Pull complete
b4624b3efe06: Pull complete
d42beb8ded59: Pull complete
5badffea4c42: Pull complete
6107652a946b: Pull complete
1b31669dbe65: Pull complete
4d884b22dc63: Pull complete
cee72f2b293c: Pull complete
33323ef67397: Pull complete
6673677e2f45: Pull complete
564a2f62e20f: Pull complete
e1edc18db05f: Pull complete
422f87ba9e77: Pull complete
Digest: sha256:d957946c3a6470ecb3c2d41bea7fba38ad07f10b2f3fe13bb711f42fba8c5022
Status: Downloaded newer image for mariadb:latest
Creating student_db_1
Pulling wordpress (wordpress:latest)...
latest: Pulling from library/wordpress
6ae821421a7d: Already exists
08f3d19635b0: Pull complete
dc8a54b8000b: Extracting [=====> ] 66.29MB/67.44MB
b2c1d103db99: Download complete
edfa752aa38a: Download complete
583d37cbf2f0: Download complete
c7846a240c1d: Download complete
d8f9f0fd02fe: Download complete
01d43e56770d: Download complete
dbe439e2caf9: Download complete
3de30e1f5211: Download complete
209dd35ef060: Download complete
3d97847926b1: Download complete
e2dfaa5afe2c: Download complete
c39665b60f96: Download complete
f36ef5821516: Download complete
c96693ff91d0: Download complete
2e00dc04ad85: Download complete

```

## 2. Check the progress of the container deployment

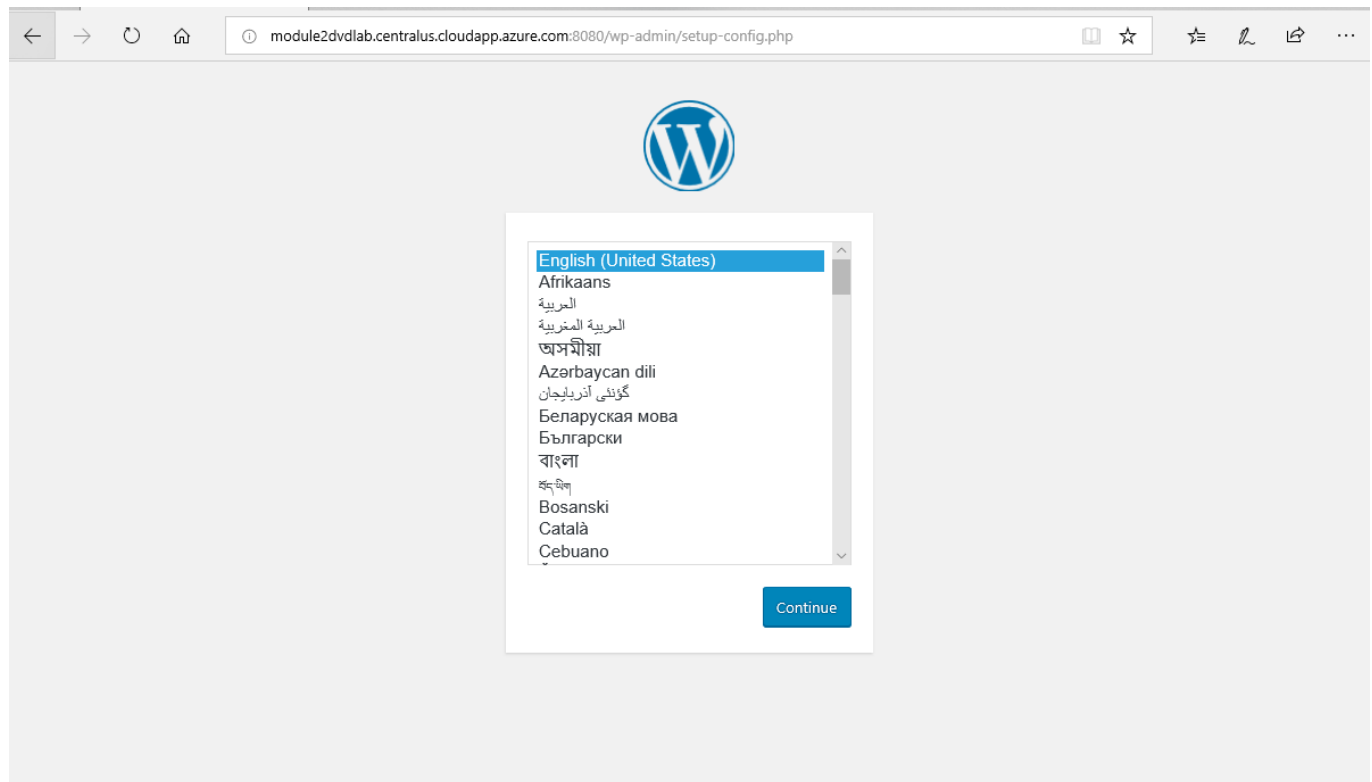
```
docker ps
```

```

student@MyDockerVM:~$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS               NAMES
6b1b045b7841        wordpress           "docker-entrypoint.s..." 29 seconds ago      Up 28 seconds      0.0.0.0:8080->80/tcp student_wordpress_1
a8867a79f84f        mariadb             "docker-entrypoint.s..." 55 seconds ago      Up 54 seconds      3306/tcp            student_db_1
6ff89b0fce22        nginx               "nginx -g 'daemon of..." About an hour ago    Up About an hour    0.0.0.0:80->80/tcp   jolly_neumann

```

## 3. start Microsoft Edge and browse to the port 8080 on the target host using the same URL you used in the previous exercise. Verify that Microsoft Edge displays the initial Wordpress setup page.



4. Type exit to terminate the SSH session and then close the Cloud Shell pane.

Result: Once you have completed this exercise, you have successfully implemented a multi-container application by using Docker Compose.